IN THE SPECIFICATION

Please rewrite the paragraph at page 6, lines 7-12 as follows:

Various results data are then provided to, and thereafter output from a preprocessing system ~~2121~~ 2125, and are in turn acted upon in various post processing functions 2130 employing user defined control variables 2120, resulting in processed data (Results data) 2135. The post-processed data is then exported and/or imported at 2140 as necessary for further processing by the system 2145, also in accordance with control variables 2120. After processing has been completed, the various processed data is converted for display at 2150 on a display device 2155.

Please rewrite the paragraph at page 8, lines 1-18 as follows:

In this simple, fixed oscilloscope apparatus, all objects, either input or output, are represented as objects on the border of the web diagram, while the connections between the various objects are represented as connections within the processing web. The placement of these processing elements along the outer border is merely a graphical representation convention for placing inputs on one edge and outputs on another. However, this graphical display has no affect on a working system. This particular processing web graphic is a representation of an older oscilloscope design where each processing element receives predefined inputs as a combination of dynamic inputs, static inputs, parameters, and in some cases prior trace views and all function on a single system clock, all at the same speed and refresh rate. Thus, in this particular representation, there is no provision for the chaining of various processing elements, or functioning processing elements at different speeds, and indeed each processing element shown in Fig. 1 produces a predetermined output at a predetermined time. It is therefore not possible to

modify the processing of the oscilloscope to provide different desirable outputs. While other LeCroy oscilloscopes have allowed for certain chaining and different processing speeds within a system. ~~However~~, the ability to utilize these features was limited to a number of predetermined scenarios. As will be noted below, in accordance with the invention, these limitations have been removed, and a more general solution is provided that allows for a completely dynamic set of scenarios with differing data rates and differing requirements on different chains of processors.

Please rewrite the paragraph at page 11, lines 7-11 as follows:

1. All Processor objects must meet the requirements of a generic processor object model. The fundamental requirements of such a generic processor is that the Processor has 0 or more input pins, 0 or more output pins and 0 or more update pins. Obviously, in order to be of any use, any given Processor object must have at least 1 input ~~and~~ or at least ~~one~~ 1 output pin of some particular type.

Please rewrite the paragraph at page 12, line 5 – page 13, line 4 as follows:

In Fig. 2, an Acquisition Board processor class 210 does not have any inputs ~~216~~ and has 4 outputs 216 that produce waveforms (1 for each acquisition channel on the board, i.e. C1, C2, C3 and C4 of Fig. 1). A Waveform Averager processor class 220 has 1 input 222 that comprises an input waveform and 1 output 226 that produces a waveform. Furthermore, it includes an update pin 224 that explicitly controls when the waveform produced from the output is updated with respect to the waveforms seen at the input (i.e., precisely the timing of when the processing in the waveform averager is to be implemented). A Waveform Adder processor class 230 has two inputs 232 that comprise waveforms and one output 236 that produces a waveform.

Waveform Adder processor class 230 does not have any update pin, and therefore, the waveform produced by its output is always continuously updated in real time with respect to the waveforms seen at its inputs. A Trace Renderer processor class 240 has 1 input 242 that comprises a waveform and 1 update pin 244 that specifies explicitly when the Trace Renderer should sample the waveforms seen at its input. It does not have any result outputs, but rather instead of producing any further results, it draws in a graphics window a representation of the waveforms seen at its input. The final two examples shown in Fig. 2 illustrate other types of results (namely parameters and histograms). An Amplitude processor class 250 comprises waveforms from its input 252 and produces parameter results at its output 256. A Parameter Histogrammer processor class 260 comprises parameter results at its input 262 and produces histogram results at its output 266, with an update pin 264 resetting the histogram and beginning the data accumulation process. Thus, because it has an update pin, it is explicitly controlled via the update pin to know when it should receive parameter results at its input and update the histogram results produced at its output. A further description of update pins and the control thereof will be discussed below.

Please rewrite the paragraph at page 15, line 18 – page 16, line 19 as follows:

As is clear from this description, it is necessary to manage the implementation of the various update ~~pints~~ pins to insure that proper data is received at each node in the system and that the sequence of update pin implementation insures proper functioning of the system. A Processing Web Manager software object is therefore responsible for managing the Processing Web and updating subsets of Processor objects in response to requests and events from other

software objects. This Processing Web Manager object maintains a list of all of the processor objects and analyzes the inter-connection paths and processor objects themselves to determine which processors should be updated and in which order based upon which events. In order to determine when the processors should be updated, the Processing Web Manager object reacts to 3 different events or requests: (1) an indication that new results are available at an output of some processor object (New Results Available), (2) a request for a synchronization of all or some of the processors on the processing web such as when a particular function (SynchronizeWeb) is to be performed that would require reprocessing of data by a number of processing objects and (3) a change in the definition of one of the processing objects on the processing web (Definition Changed). The first two events (1) New Results Available and (2) Synchronize Web[[)]] are posted by other fundamental system manager objects. A Result Source Manager (or Acquisition Manager) tells the Processing Web Manager when new source (acquisition) results are available via the NewResultsAvailable request. A Result Sink Manager (or Display Manager) tells the Processing Web Manager when it wishes to take a 'synchronized snapshot' of the renderer (or result sink) processor objects via the SynchronizeWeb request. Such a snapshot may be employed upon a downstream request for data, resulting in clocking through of upstream data. The DefinitionChanged request is used by any of the included processor objects to notify the Processing Web Manager when a processor's definition has changed in order to update all processing objects in response to the processor's definition change.

Please rewrite the paragraph at page 24, lines 2-5 as follows:

Thus, ~~as is shown~~ in Fig. 10A shows by way of example only, attempts to drag a connection between 'C1' and 'In'. The C1 pin is an integer waveform type, the FFT input 'In' is a floating-point waveform type.

Please rewrite the paragraph at page 24, lines 15-19 as follows:

Finally, as is shown in Fig. 11C, if the PWEditor is instructed to display the hidden adapters, the complete picture of how the connection was made is displayed. Thus, first integer waveform data ~~110~~ 1110 is converted to floating-point waveform data at ~~120~~ 1120, then a second adapter converts ~~130~~ 1130 the floating-point waveform data into multiple parameter values (scalars) which may then be fed to the histogrammer 1140 to generate a histogram output 1150.

Please rewrite the paragraph at page 25, lines 17-21 as follows:

Thus, in addition to providing the five objects of Fig. 13, the waveform amplitude calculator 1450 and display thereof 1460 were dragged into the processing web, and connected as shown in Fig 14. Thus, the parameter measurement is performed at 1450, and the numeric parameter output is displayed at 1460. The output ~~form~~ from node ~~1110(C_1)~~ 1310(C_1) is forwarded to two locations, one of display (1330) and one for further processing (1450, 1460).

Please rewrite the paragraph at page 26, line 20 – page 27, line 10 as follows:

If however, at step 1610 no further components in the web are available, and therefore the inquiry is answered in the negative, control then passes to step 1635 where the processing

web manager is asked for the first connection in the web. Control then passes to step 1640 where it is determined whether such a connection has been provided. If so, control passes to step 1645 where the pins defined by the connection are connected by a line, preferably colored based upon the data type flowing through the connection. Of course, any other graphical designation may be employed for a line, such as texture, crosshatch, a symbol or the like. After this connection has been graphically illustrated, control passes to step 1650 where the processing web manager is asked for a next connection in the web. If such a connection exists, as in step 1640, the inquiry will be answered in the affirmative and control will pass to step 1645, thereby repeating this procedure. If at step 1640 it is determined that no further connections in the web ~~exists~~ exist and the entire web has been graphically illustrated, the inquiry at step 1640 will be answered in a negative and the process will end. Therefore, in accordance with the procedure set forth in ~~the phase~~ Fig. 16, a graphical representation can be displayed to a user of an existing web.